RECEIVED
CENTRAL FAX CENTER

APR 0 9 2007

Attorney's Docket No.: 07844-499001
Client's Ref. No.: P463

OFFICIAL COMMUNICATION FACSIMILE:

OFFICIAL FAX NO: (571) 273-8300

Number of pages including this page 28

Applicant : Jonathan Shekter
Serial No. : 10/080,525
Filed : February 21, 2002

Art Unit : 2628
Examiner : Peter-Anthony Pappas

Title : COMPOSITE RENDERING 3-D GRAPHICAL OBJECTS

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

A Brief on Appeal dated **April 9, 2007** is attached.

Respectfully submitted,

Date: April 9, 2007

John F. Horvath
Reg. No. 47,663

Fish & Richardson P.C.
One Congress Plaza, Suite 810
111 Congress Avenue
Austin, TX 78701
Telephone: (512) 472-5070
Fax: (512) 320-8935

11017723.doc

Attorney's Docket No.: 07844-499001 / P463

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant	: Jonathan Shekter	Art Unit	: 2628
Serial No.	: 10/080,525	Examiner	: Peter-Anthony Pappas
Filed	: February 21, 2002	Conf. No.	: 8647
Title	: COMPOSITE RENDERING 3-D GRAPHICAL OBJECTS		

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

BRIEF ON APPEAL

(1) Real Party in Interest

Adobe Systems Incorporated

(2) Related Appeals and Interferences

None

(3) Status of Claims

Claims 6-45 are pending in the application. Claims 1-5 have been cancelled. Claims 34, 37 and 39 have been allowed. Claims 14, 17 and 19 have been objected to as depending from a rejected base claim, but as otherwise allowable if written in independent form. Claims 6-13, 15, 16, 18, 20-33, 35, 36, 38 and 40-45 have been rejected. Applicant appeals the rejection of claims 6-13, 15, 16, 18, 20-33, 35, 36, 38 and 40-45 and the objection to claims 14, 17 and 19.

(4) Status of Amendments

None

(5) Summary of Claimed Subject Matter

The application discloses "a 3-D image compositing system that allows 3-D objects to be separately rendered and combined together in a realistic-looking composite image or scene having all of the 3-D image processing effects that add realism to the scene such as anti-

CERTIFICATE OF TRANSMISSION BY FACSIMILE

I hereby certify that this correspondence is being transmitted by facsimile to the Patent and Trademark Office on the date indicated below.

Date of Transmission: April 9, 2007

Signature: [signature]

Kathy Gilbert
Typed or Printed Name of Person Signing Certificate

or scene having all of the 3-D image processing effects that add realism to the scene such as anti-aliasing, motion-blur, and depth of field effects." *Application* at 2:9-12.[1] The disclosed system improves upon prior art systems in that it allows separately rendered 3-D objects to be subsequently composited or combined together in a 2-D scene in a way that preserves anti-aliasing, motion-blur, and depth of field corrections. This allows independently rendered objects to be easily "added and removed from a scene without having to re-render the entire scene," and without losing the rendering effects that add realism to the scene. *Id.* at 1:23-25. It also "allow[s] for objects to be separately created, rendered, and realistically used and re-used in a multitude of different scenes." *Id.* at 1:25-26. The disclosed system "can be used as an ordinary 3-D renderer to render 3-D objects together in a scene, or as a 3-D composite renderer to combine separately rendered 3-D objects into a scene and to correctly include anti-alias, motion-blur, and depth-of-field effects at the intersections of objects within the scene." *Id.* at 2:14-17.

Prior to applicant's invention, prior art image compositing systems were unable "to incorporate all of the image processing effects that would allow realistic looking 3-D composited scenes to be created." *Id.* at 1:28-2:1.[2] That's because prior art systems that "combined anti-aliasing, motion-blur, and depth-of-field effects, *require[d]* the individual components of the scene to be *rendered together in the same rendering step.*" *Id.* at 1:15-18 (emphasis added). Thus, separately rendered 3-D objects could not be added to a scene without losing all of the effects that add realism to the scene. Nor could individual 3-D objects be removed from the scene.

The image compositing system disclosed in the application allows 3-D objects to be independently rendered and composited to a 2-D scene while preserving realism-adding effects by first rendering the 3-D objects "to a motion buffer or M-buffer," which can be subsequently resolved in a way that preserves anti-aliasing, motion-blur and depth-of-field blur effects. *Id.* at 4:16-19. To anti-alias and depth-of-field blur 3-D objects rendered to an M-buffer, particularly when they intersect, "information about the surface geometry" of the objects is stored in the M-buffer. *Id.* at 8:20-22 and 14:21-23. To motion-blur 3-D objects that have been rendered to an

---

1 Notations of the form X:Y-Z refer to page X, lines Y-Z when reference is made to the application, and to column X, lines Y-Z when reference is made to a patent

2 Notations of the form X:Y-Z:W refer to page X, line Y through page Z, line W when reference is made to the application, and to column X, line Y through page Z, line W when reference is made to a patent.

M-buffer, "the z-component of velocity or *dz/dt* of each object moving in the z-direction" is stored in the M-buffer. *Id.* at 9:13-16. Individual, separately rendered 3-D objects can be composited to a 2-D scene because rendered object data stored in one M-buffer can be added to rendered object data stored in another M-buffer, such that when the resulting M-buffer is resolved, its contents are correctly anti-aliased, motion-blurred, and depth-of-field blurred. *See id.* at 16:24-25.

Independent claims 6, 9, 22, 26, 29 and 42 are currently pending in the application and on appeal. Claims 6 recites a method for creating a motion buffer, while claims 9 and 22 recite a method for compositing one or more scan-converted 3-D objects to a 2-D scene utilizing a motion buffer that is (claim 9) received by an application, or (claim 22) created and filled by rendering all non-simple 3-D object clusters in the scene. Claims 26, 29 and 42 are Beauregard claims that recite computer program products comprising instructions operable to cause a programmable processor to perform the methods recited in claims 6, 9 and 22, respectively.

Per independent claim 6, a method for creating a motion buffer to store the local properties of one or more scan-converted 3-D objects comprises "receiving one or more 3-D objects," each of which "comprises one or more object primitives." *See id.* at 3:10-12 (Fig. 1 at 102); *see also id.* at 10:6-7 (Fig. 4 at 401-409). Each 3-D object's one or more object primitives are then scan-converted "into a plurality of pixel fragments corresponding to a plurality of pixels in a 2-D scene, wherein each pixel fragment is configured to store the local properties of a scan-converted object primitive, including the object primitive's local color, depth, coverage, transfer mode, rate of change of depth with time, and surface geometry information, wherein the surface geometry information comprises spatial information about the object primitive's surface." *See id.* at 10:7-10 (Fig. 4 at 405). This is done by rendering the object primitives "to a motion buffer or M-buffer." *Id.* at 4:16-17 (Fig. 1 at 107). The M-buffer "can be implemented as an array of linked lists 210 having a one-to-one correspondence with the pixels in the screen space." *Id.* at 5:3-5 (Fig. 2 at 210). Each linked list "can contain from zero to *n* pixel fragments 220, where *n* is typically the number of objects in the non-simple object cluster that is rendered into [the] M-buffer." *Id.* at 5:12-14. The pixel fragments can "store information about the local properties of the objects that are rendered to the output buffer pixel." *Id.* at 5:25-27. This information can include "the object primitive's coverage 320 of the output buffer pixel; the object primitive's

depth 330 or *z*-position in the output buffer pixel; the object primitive's *z*-component of velocity or *dz/dt* 340 in the output buffer pixel; the orientation *dz/dx* 350 and *dz/dy* 360 of the surface of the object primitive relative to the plane of the output buffer pixel; the object primitive's color 370 and [the object primitive's] transfer mode 380." *Id.* at 5:27- 6:6 (Fig. 3 at 320-380). Once rendered, the information from each of the pixel fragments is inserted "into the motion buffer for subsequent composition to the 2-D scene." *See id.* at 4:16-17 (Fig. 1 at 107); *see also id.* at10:10-15 (Fig. 4 at 406).

Independent claim 26 is a Beauregard claim reciting a computer program comprising instructions operable to cause a programmable processor to perform the steps recited in method claim 6. It is therefore supported by the same passages described above, which support claim 6, and the applicant's disclosure that "[t]he invention can be implemented in digital electronic circuitry, or in computer hardware, firmware software of in combinations of them. Apparatus of the invention can be implemented in a computer program product tangibly embodied in a machine-readable storage device for execution by a programmable processor." *Id.* at 15:13-16.

Per independent claim 9, a method for compositing one or more scan-converted 3-D objects to a 2-D scene comprises "receiving a motion buffer, the motion buffer containing the rendered local properties of the one or more scan-converted 3-D objects including each scan-converted 3-D object's color, depth, coverage, transfer mode, rate of change of depth with time, and surface geometry information, wherein the surface geometry information comprises spatial information about the scan-converted object's surface." A method for generating a motion buffer with the properties recited in claim 9 is discussed above in reference to claim 6. Claim 9 recites receiving such a motion buffer rather than generating it. As disclosed in the specification, "[i]n step 207 [sic, 107] some of the object primitives in the non-simple object cluster may have been previously rendered to an M-buffer. These object primitives can be read from that M-buffer and written directly to M-buffer 200 without being re-rendered by a 3-D rendering algorithm." *Id.* at 4:25-28 (Fig. 1 at 107). Once a motion buffer is received by an application, it can be resolved "by using the information stored in the motion buffer to composite the one or more scan-converted 3-D objects to the 2-D scene." *See id.* at 4:16-19 (Fig. 1 at 108); *see also id.* at 13:6-20 (Fig. 6).

Independent claim 29 is a Beauregard claim reciting a computer program comprising instructions operable to cause a programmable processor to perform the steps recited in method claim 9. It is therefore supported by the same passages described above, which support claim 9, and the applicant's disclosure that "[t]he invention can be implemented in digital electronic circuitry, or in computer hardware, firmware software of in combinations of them. Apparatus of the invention can be implemented in a computer program product tangibly embodied in a machine-readable storage device for execution by a programmable processor." *Id.* at 15:13-16.

Per independent claim 22, a method for rendering a plurality of scan-converted 3-D objects to a 2-D scene comprises "splitting the plurality of scan-converted 3-D objects into one or more object clusters." *See id.* at 3:13-16 (Fig. 1 at 103). The object clusters are tested for simplicity, and all non-simple object clusters are rendered "to a motion buffer, the motion buffer containing the rendered local properties of the one or more scan-converted 3-D objects including each scan-converted 3-D object's color, depth, coverage, transfer mode, rate of change of depth and surface geometry information." *See id.* at 4:4-17 (Fig. 1 at 106-107). A suitable method for generating a motion buffer having the properties recited in claim 22 is discussed above in reference to claim 6. Once created, the motion buffer is resolved "to composite the non-simple object clusters to the 2-D scene." *See id.* at 4:16-19 (Fig. 1 at 108); *see also id.* at 13:6-20 (Fig. 6).

Independent claim 42 is a Beauregard claim reciting a computer program comprising instructions operable to cause a programmable processor to perform the steps recited in method claim 22. It is therefore supported by the same passages described above, which support claim 22, and the applicant's disclosure that "[t]he invention can be implemented in digital electronic circuitry, or in computer hardware, firmware software of in combinations of them. Apparatus of the invention can be implemented in a computer program product tangibly embodied in a machine-readable storage device for execution by a programmable processor." Id. at 15:13-16.

(6) Grounds of Rejection to be Reviewed on Appeal

(a) Whether claims 6-12, 15, 16, 18, 22-24, 26-33, 35, 36, 38 and 42-44 are obvious in view of the combination of U.S. Patent No. 5,990,904 to Griffin ("Griffin"), U.S. Patent No. 5,809,210 to Pearce et al. ("Pearce"), and U.S. Patent Publication No. 2002 / 0097241 A1 to McCormack et al. ("McCormack").

(b) Whether claims 13, 20, 21, 40 and 41 are obvious in view of the combination of Griffin, Pearce, McCormack and U.S. Patent No. 6,426,755 to Deering ("Deering").

(c) Whether claims 25 and 45 are obvious in view of the combination of Griffin, Pearce, McCormack and the treatise written by Foley et al. and entitled Computer Graphics: Principle and Practice ("Foley").

(7) Argument

(a) Claims 6-12, 15, 16, 18, 22-24, 26-33, 35, 36, 38 and 42-44

Independent claims 6 and 22 recite methods comprising creating motion buffers to store the local properties of scan-converted 3-D objects, including each scan-converted 3-D objects "color, depth, coverage, transfer mode, rate of change of depth and surface geometry information." Claim 9 recites a method comprising receiving a motion buffer that includes the local properties of scan-converted 3-D objects, including each scan-converted 3-D objects "color, depth, coverage, transfer mode, rate of change of depth and surface geometry information." As explained above, motion buffers having such properties are needed in order to add and remove individual 3-D objects that are composited to a 2-D scene without having to re-render the entire scene, while preserving the anti-aliasing, motion-blurring, and depth-of-field blurring effects that add realism to the 2-D scene. The Examiner rejected each of claims 6-12, 15, 16, 18, 22-24, 26-33, 35, 36, 38 and 42-44 as obvious in view of the combination of Griffin, Pearce and McCormack. *See Final Office Action* at ¶ 4. The applicant disagrees.

The Griffin patent "relates to graphics rendering systems that generate pixel fragments to perform anti-aliasing, and more specifically relates to a method and system for generating and storing pixel fragments." *Griffin* at 1:12-15. Griffin's system includes:

> a rasterizer, a pixel engine, a pixel buffer and a fragment buffer. The rasterizer receives geometric primitives and generates instances of pixel data, including color, depth, and coverage data. The pixel engine controls the transfer of the pixel data to the pixel and fragment buffers. The pixel buffer comprises an array of elements corresponding to pixel locations, and each element stores color and depth data for fully covered pixels closest to the viewpoint. The fragment buffer stores color, depth, and coverage data for partially covered pixels."

Id. at 5:13-15. Griffin's system also discloses storing an object primitive's α or translucency information, which is a form of transfer mode. *Id.* at 34:4-26. However, Griffin fails to disclose or even suggest storing an object primitive's rate of change of depth or surface geometry information. Thus, Griffin fails to disclose rendering a 3-D object to a motion buffer that stores the object's color, depth, coverage, transfer mode, rate of change of depth and surface geometry information, or receiving a motion buffer that includes such information as recited in the claims.

The Examiner admits, correctly, that Griffin "fails to explicitly teach rate of change of depth with time" and that Griffin "fail[s] to explicitly teach surface geometry information." *Final Office Action* at ¶ 5. This is significant, since the information that Griffin does disclose storing (color, depth, coverage and transfer mode) is insufficient to motion-blur, depth-of-field blur and anti-alias separately rendered objects that are composited together. As disclosed in the application, while "two or more separately rendered 3-D object primitives can be composited to an output buffer pixel corresponding to a linked-list 210 . . . provided that each fragment in linked list 210 stores at least each object primitive's object ID 310 (or transfer mode 380), coverage 320, depth 330, and color 370 . . . when all pixels are so composited, the resulting composite image is not properly anti-aliased, motion-blurred, or depth-of-focus blurred at any intersections between the two or more objects." *Application* at 8:13-19.

The McCormack patent, which the Examiner also relies upon to reject the claims as obvious, also fails to disclose storing a rendered 3-D object's rate of change of depth with time. The McCormack patent discloses "a system and method for reducing memory and processing bandwidth requirements of a computer graphics system by . . . merg[ing] selected image fragments before they reach a frame buffer." *McCormack* at ¶ 3. McCormack's system uses a fragment merge buffer to store "a predetermined number of fragment tuples. . . . Each fragment tuple includ[ing] a coverage mask, color value, depth (Z) value, and a pair of depth gradient (Z gradient) values." *Id.* at ¶ 19. McCormack uses the Z gradient values to calculate the depth or Z value of a pixel fragment "at any point in the fragment using a planar (affine) equation" that is parameterized by the stored Z gradient values. *Id.* at ¶¶ 70-72. McCormack does this since "representing [an] entire fragment with a single Z value leads to gross artifacts, as incorrectly computing which primitive is visible . . . at several sample points may lead to large changes in the color of the pixel." *Id.* at ¶ 70. Thus, McCormack uses the Z gradient information simply to

determine the visibility of pixel fragments rather than to properly anti-alias, motion-blur, or depth-of-field blur separately rendered fragments that are composited together. Regardless, McCormack fails to disclose storing a pixel fragment's rate of change of depth with time, and the Examiner does not rely on McCormack for disclosing storing this information.

Instead, the Examiner relies on and argues that Pearce discloses this limitation. *Final Office Action at ¶ 5*. According to the Examiner:

> Pearce et al. teaches that a view [sic, viewer] uses the blurriness of the object in the individual frames to make assumptions about its relative velocity and predictions about its position in subsequent frames (column 1, lines 12-19). Pearce et al. further teaches that the segments illustrated in Fig. 6 are drawn with respect to the z (depth) and t (time) axes. An increase in the z direction correlates to increasing depth. An increase in the t direction from 0.0 to 0.1 correlates to the movement from the S_{open} to the S_{closed} positions. Each of the segments illustrated in Fig. 6 are generated based upon the intersection of leading and/or trailing edges with a stationary sampling point (column 6, lines 42-57).
>
> It would have been obvious to one skilled in the art, at the time of the applicant's invention, to incorporate motion blur utilizing a coverage technique involving shutter exposure times, to improve the quality of animations, as taught by Pearce et al. into the system taught by Griffin, because Griffin teaches utilizing advanced real time animation in games (column 7, lines 1-4) and through such incorporation a higher quality of animation (i.e., more life like animation) would be able to be achieved.

Id. The applicant disagrees with the Examiner's interpretation of Pearce.

Pearce discloses a computer graphics imaging system capable of "the simulation of motion blur in computer generated imagery." *Pearce* at 1:8-10. Pearce's system operates by "analyzing the movement of tessellated representations of surfaces relative to a stationary sampling point on a pixel." *Id.* at 1 :45-48. Pearce's system first "identif[ies] the intersections between the leading and trailing edges of. . . individual polygon [segments] with [a] stationary sampling point," and then uses this information to determine "the sub-interval of exposure time where the sampling point is inside the polygon." *Id.* at 1:53-55. Pearce's system then places all of the polygon segments that intersect a given sampling point "in a list that is associated with the sampling point." *Id.* at 1:61-62. This list is then "sorted to remove portions of segments that are occluded and segments that are completely occluded." *Id.* at 1:63-65. The segments that are not removed in the sorting step "are then grouped together based upon the continuity of time

coverage," *id.* at 2:5-6, and combined "based upon the time coverages" to obtain "the shading or texture for [a] particular sampling point." *Id.* at 2:32-36. Pearce's method for simulating motion-blur is fully disclosed, for example, in steps 502 through 510 of figure 5.

Nowhere, in its description for simulating the motion-blur of a rendered 3-D object, does Pearce disclose calculating the object's rate of change of depth with time. Nor does Pearce ever disclose storing that information for any purpose, in any kind of buffer. Nonetheless, the Examiner argues that Pearce discloses calculating and storing a moving polygon segment's rate of change of depth with time in figure 11, and in the passages at 1:12-19, 3:49-58 and 6:42-57. *Final Office Action* at ¶¶ 5 and 41. Again, the applicant disagrees.

In the passage at 1:2-19, Pearce merely explains the desired effect – motion-blur – and not the means to achieve that effect. In particular, Pearce simply states that when a viewer sees a sequence of images that have been "recorded using standard video and film cameras," each of which contains a "noticeable blurring of moving objects," the viewer uses the motion-blur of the object's "to make assumptions about [the object's] relative velocity and predictions about its position in subsequent frames." *Pearce* at 1:12-19. Pearce fails to disclose in this passage, which concerns naturally occurring motion-blur of objects recorded on standard video cameras, any means for generating or simulating the motion-blur of computer generated objects, or what role generating and storing the rate of change of depth of rendered, computer generated objects plays in correctly motion-blurring the objects when they are composited to a scene. By contrast, in the instant application, the applicant discloses that "[t]o correctly motion-blur the intersections between two or more objects composited by depth or *z*-position when one or more of the objects are moving in the *z*-direction, the *z*-component of velocity or *dz/dt* of each object moving in the *z*-direction at each output buffer pixel must be stored." *Application* at 9:13-16.

In the passage at 3:49-58, upon which the Examiner also relies, Pearce discloses that "[a] pixel can have more than one screen sampling point," and that the "number of sampling points and the position of the sampling points within a pixel can vary." As explained above, Pearce uses pixel sampling points to simulate the motion blur of moving objects by calculating the exposure times or "time coverages" during which a given moving object's primitives intersect the sampling points, and then using those "time coverages" to compute a weighted average color from the color of each of the object primitives that were sampled by the sampling point. *See,*

e.g., id. at 1:45-2:36. The passage at 3:49-58 discloses that a single pixel in a given 2-D scene can be sub-sampled by a plurality of screen sampling points. As a result, the pixel's color must ultimately be determined by averaging the color information gathered at each of the pixel's sampling points. As Pearce explains, "the number and position of [pixel] sampling points 211-219 is chosen to provide a sufficient approximation of the color and/or intensity of pixel 200." *Id.* at 3:59-61. "To determine the color value of pixel 200, a weighted or unweighted average of the color values of each of the pixel sampling points 211-219 (and possibly including sample points from neighboring pixels) is determined." *Id.* at 3:66-4:2.

In the passage at 6:42-57, upon which the Examiner also relies, Pearce discusses the "dynamics" of the sorting process that is performed in step 502 of Figure 5. *Id.* at 6:32- 43. As discussed above, the sorting process is used to sort all of the polygon segments that intersect a given sampling point during a simulated shutter interval, and is based "upon the times of sampling point intersection and the depth of the polygon that produced the segment." *Id.* at 6:28-29. Pearce illustrates the sorting process in figure 6, "with reference to [a] graphical representation of an exemplary segment list." *Id.* at 6:42-44. As the Examiner correctly notes, the graph shown in figure 6 has both a z-axis and a t-axis. It graphically depicts an exemplary segment list that was generated by plotting "the intersection of leading and/or trailing edges" of a tessellated polygon segment that intersected a "stationary sampling point" within a scene. *Id.* at 6:49-5 1. For example, the segment $A_s - A_e$ that is shown in figure 6 was generated from the "movement of [a] polygon A through [a] sampling point," and "identif[ies] the intersection of the leading edge of polygon A with the sampling point and the intersection of the trailing edge of polygon A with the sampling point, respectively." *Id.* at 6:51-57. The Pearce system "record[s] the sequential appearance of each of the individual start and end points" of each polygon segment that intersects a sampling point. *Id.* at 6:64-66. Thus, Pearce records no more than a polygon segment's depth or z-position at the time it initially and finally intersects the sampling point. Nowhere, does Pearce disclose recording the polygon segment's rate of change of depth with time, or using that information to determine the polygon segment's motion-blur.

Finally, as noted above, the Examiner relies on figure 11 of Pearce for disclosing storing in a motion buffer the "rate of change of depth with time" limitation that is recited in rejected claims 6, 9 and 22. But Figure 11 is merely "a block diagram of a computer useful for

implementing elements of the present invention." *Id. at 2:64-65.* While the computer shown in figure 11 has a memory for storing programs and other information, nothing in the disclosure of figure 11 or in the discussion of that figure indicates that a scan-converted 3-D object primitive's "rate of change of depth with time" information is stored in the memory. Nor, as explained above, does any other passage in Pearce disclose storing this information in a buffer for any purpose. Nor do any of the other references cited by the Examiner.

For the reasons noted above, the Examiner has failed to establish a *prima facie* case that any of claims 6, 9 and 22 are obvious. The claims should be allowed to issue for at least this reason. *In re Oetiker,* 977 F.2d at 1445 (Fed. Cir. 1992). Moreover, claims 7 and 8 depend from and contain the same limitations as claim 6, while claims 10-12, 15, 16 and 18 depend from and contain the same limitations as claim 9, and claims 23 and 24 depend from and contain all the limitations of claim 22. These claims are therefore patentable over the combination of Griffin, Pearce and McCormack for at least the same reason that claims 6, 9 and 22 are patentable over that combination. Finally, claims 26-28, 29-41 and 42-44 are Beauregard claims that recite computer program products comprising instructions operable to implement the methods recited in claims 6-8, 9-21 and 22-24, respectively. They are therefore patentable over the Griffin, Pearce and McCormack combination for at least the same reason.

(b) Claims 13, 20, 21, 40 and 41

Claims 13, 20 and 21 depend from and contain all the limitations of independent claim 9, while claims 40 and 41 depend from and contain all the limitations of claim 29. The Examiner rejected each of claims 13, 20, 21, 40 and 41 as obvious in view of the combination of Griffin, Pearce, McCormack and Deering. *Final Office Action* at ¶ 31. The Examiner relies upon the Griffin, Pearce and McCormack reference to disclose a motion buffer containing "the rendered local properties of . . . one or more scan-converted 3-D objects including each scan-converted 3-D object's color, depth, coverage, transfer mode, rate of change of depth with time, and surface geometry information, wherein the surface geometry information comprises spatial information about the scan-converted object's surface." *Id.* However, as explained above, this combination of references fails to disclose such a buffer. Deering also fails to disclose such a buffer.

Deering discloses a "graphics system[] that render[s] realistic images based on three-dimensional graphics data." *Id.* at 1:6-9. Deering's system includes a memory buffer for storing the local properties of scan-converted 3-D objects. Deering renders graphical 3-D objects by "calculat[ing] the z and color information (which may include alpha or other depth of field information values) . . . and stor[ing] the data into [a] sample buffer 162." *Id.* at 14:57-60. Deering, however, fails to disclose calculating or storing a 3-D object's rate of change of depth with time information, and the Examiner does not rely on Deering for such disclosure. Since neither Deering, Griffin, Pearce, nor McCormack disclose rendering a 3-D object to a motion buffer in which the object's rate of change of depth with time information is stored, claims 13, 20, 21, 40 and 41 are patentable over the combination of these references for at least this reason.

(c) Claims 25 and 45

Claim 25 depends from and contains all the limitations of independent claim 22, while claim 45 depends from and contains all the limitations of independent claim 42. The Examiner rejected each of claims 25 and 45 as obvious in view of the combination of Griffin, Pearce, McCormack and Foley. *Final Office Action* at ¶ 37. The Examiner relies upon the Griffin, Pearce and McCormack reference to disclose a motion buffer containing "the rendered local properties of . . . one or more scan-converted 3-D objects including each scan-converted 3-D object's color, depth, coverage, transfer mode, rate of change of depth with time, and surface geometry information, wherein the surface geometry information comprises spatial information about the scan-converted object's surface." *Id.* However, as explained above, this combination of references fails to disclose such a buffer. Foley, too, fails to disclose such a buffer. Consequently, claims 25 and 45 are patentable over the combination of Griffin, Pearce, McCormack and Foley for at least this reason.

Moreover claims 25 and 45 recite a method and a computer program product comprising instructions operable to perform a method of "adding to the contents of [a] motion buffer the contents of a second motion buffer containing one or more separately rendered 3-D objects before resolving the motion buffer." The Examiner relies on Foley to meet this limitation because "Foley et al. teaches improving rasterization time through the use of parallelism, in which graphics data is processed in parallel by a plurality of graphic processing hardware." *Id.* at ¶ 38. The applicant submits that running a plurality of rendering algorithms in parallel neither

RECEIVED
CENTRAL FAX CENTER
APR 0 9 2007

discloses, nor suggests, "adding to the contents of [one] motion buffer the contents of a second motion buffer containing one or more separately rendered 3-D objects." The addition operation is inherently serial, not parallel. One cannot add the contents of one buffer to another buffer in parallel. Consequently, claims 25 and 45 are patentable over the combination of Griffin, Pearce, McCormack and Foley for at least this reason, as well.

Please apply the $500 brief fee and any other applicable charges or credits to Deposit Account No. 06-1050.

Respectfully submitted,

Date: 4/9/07

John F. Horvath
Reg. No. 47,663

Customer No. 21876

11017226

RECEIVED
CENTRAL FAX CENTER
APR 0 9 2007

Appendix of Claims

1-5. Cancelled.

6. (Previously Presented) A method for creating a motion buffer to store the local properties of one or more scan-converted 3-D objects, comprising:

receiving one or more 3-D objects, wherein each 3-D object comprises one or more object primitives;

scan-converting each 3-D object's one or more object primitives into a plurality of pixel fragments corresponding to a plurality of pixels in a 2-D scene, wherein each pixel fragment is configured to store the local properties of a scan-converted object primitive including the object primitive's local color, depth, coverage, transfer mode, rate of change of depth with time, and surface geometry information, wherein the surface geometry information comprises spatial information about the object primitive's surface; and

inserting each of the pixel fragments into the motion buffer for subsequent composition to the 2-D scene.

7. (Original) The method of claim 6, further comprising inserting each of the pixel fragments into the motion buffer in depth sorted order.

8. (Original) The method of claim 6, further comprising storing the motion buffer as a plurality of linked lists corresponding to a plurality of pixels in the 2-D scene, wherein each link in a linked list comprises a pixel fragment having a pointer to the next pixel fragment, if any, in the linked list.

9. (Previously Presented) A method for compositing one or more scan-converted 3-D objects to a 2-D scene, comprising:

receiving a motion buffer, the motion buffer containing the rendered local properties of the one or more scan-converted 3-D objects including each scan-converted 3-D object's color, depth, coverage, transfer mode, rate of change of depth with time, and surface

geometry information, wherein the surface geometry information comprises spatial information about the scan-converted object's surface; and

resolving the motion buffer by using the information stored in the motion buffer to composite the one or more scan-converted 3-D objects to the 2-D scene.

10. (Previously Presented) The method of claim 9, wherein resolving the motion buffer further comprises blending, on a per pixel basis and in depth sorted order, the color of each of the one or more 3-D objects to the color in the 2-D scene using the transfer mode of each of the one or more 3-D objects.

11. (Previously Presented) The method of claim 9, wherein the motion buffer contains surface geometry information for each of the one or more 3-D objects and resolving the motion buffer further comprises using the surface geometry information to anti-alias the one or more 3-D objects composited to the 2-D scene.

12. (Original) The method of claim 11, wherein two or more of the 3-D objects intersect over an output buffer pixel in the 2-D scene, further comprising:

determining the number of regions in the output buffer pixel in which the one or more intersecting 3-D objects are uniquely layered, and the relative coverage of each uniquely layered region;

determining a blended color for each uniquely layered region by blending in depth sorted order the color of each of the one or more 3-D objects with the color of the output buffer pixel according to each 3-D object's transfer mode; and

painting the output buffer pixel with a weighted average of the blended colors determined for each uniquely layered region, wherein the weight assigned to the blended color of a uniquely layered region is determined by the relative coverage of that region.

13. (Previously Presented) The method of claim 9, wherein the motion buffer contains surface geometry information for the one or more 3-D objects and resolving

the motion buffer further comprises using the surface geometry information to depth-of-field blur the one or more 3-D objects composited to the 2-D scene.

14. (Previously Presented) The method of claim 13, further comprising:

using the depth and surface geometry information for the one or more 3-D objects to extend, on an output buffer pixel basis, the surfaces of the one or more 3-D objects into an extended output buffer pixel;

determining whether the extended surfaces of two or more of the 3-D objects intersect over the extended output buffer pixel; and

blending the colors of the one or more 3-D objects with the color of the output buffer pixel as if two or more of the 3-D objects intersected over the output buffer pixel whenever the extended surfaces of two or more of the 3-D objects intersect over the extended output buffer pixel.

15. (Previously Presented) The method of claim 9, wherein the motion buffer contains surface geometry information for the one or more 3-D objects and resolving the motion buffer further comprises using the surface geometry information to anti-alias and depth-of-field blur the one or more 3-D objects composited to the 2-D scene.

16. (Previously Presented) The method of claim 9, wherein the motion buffer contains the rate of change of depth for each of the one or more 3-D objects, and resolving the motion buffer further comprises using the rate of change of depth for each of the one or more 3-D objects to motion-blur the one or more 3-D objects composited to the 2-D scene.

17. (Original) The method of claim 16, wherein the surfaces of two or more of the 3-D objects pass through each other over an output buffer pixel in the 2-D scene during a shutter interval, further comprising:

determining the number of time periods during the shutter interval in which the one or more 3-D objects are uniquely layered, and the duration of each uniquely layered time period;

determining a blended color for each uniquely layered time period by blending in depth sorted order the color of each of the one or more 3-D objects with the color of the output buffer pixel according to each of the one or more 3-D objects' transfer modes; and

painting the output buffer pixel with a weighted average of the blended colors for each uniquely layered time period, wherein the weight assigned to the blended color of a uniquely layered time period is determined by the duration of that time period.

18. (Previously Presented) The method of claim 9, wherein the motion buffer contains the rate of change of depth and surface geometry information for the one or more 3-D objects, and resolving the motion buffer further comprises using the rate of change of depth and surface geometry information for the one or more 3-D objects to anti-alias and motion-blur the one or more 3-D objects composited to the 2-D scene.

19. (Original) The method of claim 18, wherein the surfaces of two or more of the 3-D objects intersect and pass through each other over an output buffer pixel in the 2-D scene during a shutter interval, further comprising:

dividing the area of the output buffer pixel and the shutter interval into a number of uniquely layered space-time regions, wherein for each uniquely layered space-time region the surfaces of the one or more 3-D objects are uniquely layered over a portion of the output buffer pixel for a portion of the shutter interval;

determining the number and volume of each uniquely layered space-time region, wherein the volume of a uniquely layered space-time region is calculated from the portion of the output buffer pixel and the portion of the shutter interval occupied by the space-time region;

determining a blended color for each uniquely layered space-time region by blending in depth sorted order the color of each of the one or more 3-D objects stored in the motion buffer with the color of the output buffer pixel according to each object's transfer mode; and

painting the output buffer pixel with a weighted average of the blended colors for each uniquely layered space-time region, wherein the weight assigned to the blended color of a

uniquely layered space-time region is determined by the volume of that uniquely layered space-time region.

20. (Previously Presented) The method of claim 9, wherein the motion buffer contains the rate of change of depth and surface geometry information for the one or more 3-D objects, and resolving the motion buffer further comprises using the rate of change of depth and surface geometry information for the one or more 3-D objects to motion-blur and depth-of-field blur the one or more 3-D objects while composited to the 2-D scene.

21. (Previously Presented) The method of claim 9, wherein the motion buffer contains the rate of change of depth and surface geometry information for the one or more 3-D objects, and resolving the motion buffer further comprises using the rate of change of depth and surface geometry information for the one or more 3-D objects to anti-alias, motion-blur and depth-of-field blur the one or more 3-D objects composited to the 2-D scene.

22. (Previously Presented) A method for rendering a plurality of scan-converted 3-D objects to a 2-D scene, comprising:

splitting the plurality of scan-converted 3-D objects into one or more object clusters;

rendering all non-simple object clusters to a motion buffer, the motion buffer containing the rendered local properties of the one or more scan-converted 3-D objects including each scan-converted 3-D object's color, depth, coverage, transfer mode, rate of change of depth and surface geometry information; and

resolving the motion buffer to composite the non-simple object clusters to the 2-D scene.

23. (Previously Presented) The method of claim 22, further comprising rendering all simple object clusters directly to the 2-D scene.

24. (Previously Presented) The method of claim 22, further comprising rendering all simple object clusters to the motion buffer before resolving the motion buffer.

25. (Previously Presented) The method of claim 22, further comprising adding to the contents of the motion buffer the contents of a second motion buffer containing one or more separately rendered 3-D objects before resolving the motion buffer.

26. (Previously Presented) A computer program product, implemented on a machine readable medium, for creating a motion buffer to store the local properties of one or more 3-D objects, the computer program product comprising instructions operable to cause a programmable processor to:

receive one or more 3-D objects, wherein each 3-D object comprises one or more object primitives;

scan-convert each 3-D object's one or more object primitives into a plurality of pixel fragments corresponding to a plurality of pixels in a 2-D scene, wherein each pixel fragment is configured to store the local properties of a scan-converted object primitive including the object primitive's local color, depth, coverage, transfer mode, and at least one of the object primitive's rate of change of depth with time, and surface geometry information, wherein the surface geometry information comprises spatial information about the object primitive's surface; and

insert each of the pixel fragments into the motion buffer for subsequent composition to the 2-D scene.

27. (Original) The computer program product of claim 26, further comprising instructions operable to cause a programmable processor to insert each of the pixel fragments into the motion buffer in depth sorted order.

28. (Original) The computer program product of claim 26, further comprising instructions operable to cause a programmable processor to store the motion buffer as a plurality of linked lists corresponding to a plurality of pixels in the 2-D scene, wherein each link in a

linked list comprises a pixel fragment having a pointer to the next pixel fragment, if any, in the linked list.

29. (Previously Presented) A computer program product, implemented on a machine readable medium, for compositing one or more scan-converted 3-D objects to a 2-D scene, the computer program product comprising instructions operable to cause a programmable processor to:

receive a motion buffer, the motion buffer containing the rendered local properties of the one or more scan-converted 3-D objects including each scan-converted 3-D object's color, depth, coverage, transfer mode, and at least one of each scan-converted 3-D object's rate of change of depth with time, and surface geometry information, wherein the surface geometry information comprises spatial information about the scan-converted 3-D object's surface; and

resolve the motion buffer by using the information stored in the motion buffer to composite the one or more scan-converted 3-D objects to the 2-D scene.

30. (Previously Presented) The computer program product of claim 29, wherein the instructions to resolve the motion buffer further comprises instructions to blend, on a per pixel basis and in depth sorted order, the color of each of the one or more 3-D objects to the color in the 2-D scene using the transfer mode of each of the one or more 3-D objects.

31. (Previously Presented) The computer program product of claim 29, wherein the motion buffer contains surface geometry information for each of the one or more 3-D objects and the instructions to resolve the motion buffer further comprises instructions to use the surface geometry information to anti-alias the one or more 3-D objects composited to the 2-D scene.

32. (Original) The computer program product of claim 31, wherein two or more of the 3-D objects intersect over an output buffer pixel in the 2-D scene, further comprising instructions operable to cause a programmable processor to:

determine the number of regions in the output buffer pixel in which the one or more intersecting 3-D objects are uniquely layered, and the relative coverage of each uniquely layered region;

determine a blended color for each uniquely layered region by blending in depth sorted order the color of each of the one or more 3-D objects with the color of the output buffer pixel according to each 3-D object's transfer mode; and

paint the output buffer pixel with a weighted average of the blended colors determined for each uniquely layered region, wherein the weight assigned to the blended color of a uniquely layered region is determined by the relative coverage of that region.

33. (Previously Presented) The computer program product of claim 29, wherein the motion buffer contains surface geometry information for the one or more 3-D objects and the instructions to resolve the motion buffer further comprises instructions to depth-of-field blur the one or more 3-D objects composited to the 2-D scene.

34. (Previously Presented) A computer program product, implemented on a machine readable medium, for compositing one or more scan-converted 3-D objects to a 2-D scene, the computer program product comprising instructions operable to cause a programmable processor to:

receive a motion buffer, the motion buffer containing the rendered local properties of the one or more scan-converted 3-D objects including each scan-converted 3-D object's color, depth, coverage, transfer mode and surface geometry information; and to

resolve the motion buffer by using the information stored in the motion buffer to composite and depth-of-field blur the one or more scan-converted 3-D objects to the 2-D scene; wherein the instructions to depth-of-field blur the one or more scan-converted 3-D objects further comprises instructions to:

use the depth and surface geometry information for the one or more 3-D objects to extend, on an output buffer pixel basis, the surfaces of the one or more 3-D objects into an extended output buffer pixel;

determine whether the extended surfaces of two or more of the 3-D objects intersect over the extended output buffer pixel; and

blend the colors of the one or more 3-D objects with the color of the output buffer pixel as if two or more of the 3-D objects intersected over the output buffer pixel whenever the extended surfaces of two or more of the 3-D objects intersect over the extended output buffer pixel.

35. (Previously Presented) The computer program product of claim 29, wherein the motion buffer contains the surface geometry information for the one or more 3-D objects and the instructions to resolve the motion buffer further comprises instructions to use the surface geometry information to anti-alias and depth-of-field blur the one or more 3-D objects composited to the 2-D scene.

36. (Previously Presented) The computer program product of claim 29, wherein the motion buffer contains the rate of change of depth for each of the one or more 3-D objects, and the instructions to resolve the motion buffer further comprises instructions to use the rate of change of depth for each of the one or more 3-D objects to motion-blur the one or more 3-D objects composited to the 2-D scene.

37. (Previously Presented) A computer program product, implemented on a machine readable medium, for compositing one or more scan-converted 3-D objects to a 2-D scene, wherein the surfaces of two or more of the 3-D objects pass through each other over an output buffer pixel in the 2-D scene during a shutter interval, the computer program product comprising instructions operable to cause a programmable processor to:

receive a motion buffer, the motion buffer containing the rendered local properties of the one or more scan-converted 3-D objects including each scan-converted 3-D object's color, depth, coverage, transfer mode and rate of change of depth; and to

resolve the motion buffer by using the information stored in the motion buffer to composite and motion-blur the one or more scan-converted 3-D objects to the 2-D scene;

wherein the instructions to motion-blur the one or more scan-converted 3-D objects further comprises instructions to:

determine the number of time periods during the shutter interval in which the one or more 3-D objects are uniquely layered, and the duration of each uniquely layered time period;

determine a blended color for each uniquely layered time period by blending in depth sorted order the color of each of the one or more 3-D objects with the color of the output buffer pixel according to each of the one or more 3-D objects' transfer modes; and

paint the output buffer pixel with a weighted average of the blended colors for each uniquely layered time period, wherein the weight assigned to the blended color of a uniquely layered time period is determined by the duration of that time period.

38. (Previously Presented) The computer program product of claim 29, wherein the motion buffer contains the rate of change of depth and surface geometry information for the one or more 3-D objects, and the instructions to resolve the motion buffer further comprises instructions to use the rate of change of depth and surface geometry information for the one or more 3-D objects to anti-alias and motion-blur the one or more 3-D objects composited to the 2-D scene.

39. (Previously Presented) A computer program product, implemented on a machine readable medium, for compositing one or more scan-converted 3-D objects to a 2-D scene, wherein the surfaces of two or more of the 3-D objects intersect and pass through each other over an output buffer pixel in the 2-D scene during a shutter interval, the computer program product comprising instructions operable to cause a programmable processor to:

receive a motion buffer, the motion buffer containing the rendered local properties of the one or more scan-converted 3-D objects including each scan-converted 3-D object's color, depth, coverage, transfer mode, rate of change of depth and surface geometry information; and to

resolve the motion buffer by using the information stored in the motion buffer to composite, anti-alias and motion-blur the one or more scan-converted 3-D objects to the 2-D scene; wherein the instruction to anti-alias and motion-blur the one or more scan-converted 3-D objects further comprises instructions to:

divide the area of the output buffer pixel and the shutter interval into a number of uniquely layered space-time regions, wherein for each uniquely layered space-time region the surfaces of the one or more 3-D objects are uniquely layered over a portion of the output buffer pixel for a portion of the shutter interval;

determine the number and volume of each uniquely layered space-time region, wherein the volume of a uniquely layered space-time region is calculated from the portion of the output buffer pixel and the portion of the shutter interval occupied by the space-time region;

determine a blended color for each uniquely layered space-time region by blending in depth sorted order the color of each of the one or more 3-D objects stored in the motion buffer with the color of the output buffer pixel according to each object's transfer mode; and

paint the output buffer pixel with a weighted average of the blended colors for each uniquely layered space-time region, wherein the weight assigned to the blended color of a uniquely layered space-time region is determined by the volume of that uniquely layered space-time region.

40. (Previously Presented) The computer program product of claim 29, wherein the motion buffer contains the rate of change of depth and surface geometry information for the one or more 3-D objects, and the instructions to resolve the motion buffer further comprises instructions to use the rate of change of depth and surface geometry information for the one or more 3-D objects to motion-blur and depth-of-field blur the one or more 3-D objects composited to the 2-D scene.

41. (Previously Presented) The computer program product of claim 29, wherein the motion buffer contains the rate of change of depth and surface geometry information for the one or more 3-D objects, and the instructions to resolve the motion buffer further comprises instructions to use the rate of change of depth and surface geometry information for the one or more 3-D objects to anti-alias, motion-blur and depth-of-field blur the one or more 3-D objects composited to the 2-D scene.

42. (Previously Presented) A computer program product, implemented on a machine readable medium, for rendering a plurality of scan-converted 3-D objects to a 2-D scene, the computer program product comprising instructions operable to cause a programmable processor to:

split the plurality of scan-converted 3-D objects into one or more object clusters;

render all non-simple object clusters to a motion buffer, the motion buffer containing the rendered local properties of the one or more scan-converted 3-D objects including each scan-converted 3-D object's color, depth, coverage, transfer mode, rate of change of depth and surface geometry information; and

resolve the motion buffer to composite the non-simple object clusters to the 2-D scene.

43. (Previously Presented) The computer program product of claim 42, further comprising instructions operable to cause the programmable processor to render all simple object clusters directly to the 2-D scene.

44. (Previously Presented) The computer program product of claim 42, further comprising instructions operable to cause the programmable processor to render all simple object clusters to the motion buffer before resolving the motion buffer.

45. (Previously Presented) The computer program product of claim 42, further comprising instructions operable to cause the programmable processor to add to the contents of the motion buffer the contents of a second motion buffer containing one or more separately rendered 3-D objects before resolving the motion buffer.

Evidence Appendix

None

Related Proceedings Appendix

None